Translation-Based Indexing for Cross-Language Retrieval

Douglas W. Oard and Funda Ertunc

College of Information Studies and
Institute for Advanced Computer Studies
University of Maryland, College Park, MD 20742, USA
oard@glue.umd.edu,
and
Department of Computer Science and
Institute for Advanced Computer Studies
University of Maryland, College Park, MD 20742,USA
efunda@cs.umd.edu

Abstract. Structured queries have proven to be an effective technique for cross-language information retrieval when evidence about translation probability is not available. Query execution time is adversely impacted, however, because the full postings list for each translation is used in the computation. This paper describes an alternative approach, translation-based indexing, that improves query-time efficiency by integrating the translation and indexing processes. Experiment results demonstrate that similar effectiveness can be achieved at a cost in indexing time that is roughly linear in the average number of known translations for each term.

1 Introduction

Use of the Internet is increasing rapidly throughout the world, with content in languages other than English now increasing far more rapidly than content in English. For example, Grefenstette found that between 1999 and 2000, English content grew by 800%, German content grew by 1500% and Spanish content grew by 1800% [2]. It is now estimated by some sources that there is more non-English than English text in the visible portion of the Web, and if present trends continue the importance to users of finding materials in languages other than that used in their query will likely continue to increase. This is the goal of Cross-Language Information Retrieval(CLIR) systems: to allow users to present a query in one language and retrieve documents that are written in a different language. Searchers who are able to read more than one language can use the results of such systems directly, formulating queries only once (in their most fluent language). Searchers without the language skills to read the documents that are found can also benefit from CLIR systems, but only if suitable human or machine translation capabilities can be provided. The widespread availability of Web-based translation services now promises at least some degree of support for using documents that are found using CLIR systems—the next challenge is to build and deploy efficient and effective CLIR systems that are compatible with the search system architectures used by high-volume Web search engines.

In CLIR, two alternative architectures have been explored:

Query translation, in which the query is translated into that language(s) in which the documents are written.

Document translation, in which the documents are translated at indexing time into the language(s) in which the queries are expected to be posed.

Query translation is the more widely studied approach, at least in part because in experimental settings it is far more efficient to translate the relatively few (perhaps 50) queries than to translate all of the documents. In high-volume production applications the reverse might be true—a substantial speedup in query processing might easily justify additional work at indexing time (particularly if only one query language is to be supported).

Regardless of which architecture is chosen, dictionary-based translation introduces three challenges:

- what to translate (e.g., word roots, words, and/or phrases),
- where to obtain the needed translation knowledge (e.g., extraction from machine readable dictionaries, construction from translation-equivalent (parallel) texts, and/or harvesting Web-accessible bilingual term lists), and
- how that translation knowledge should be used.

In this paper we adopt simple but workable approaches for the first two challenges (we translate words using a single bilingual term list found on the Web) and focus on the third challenge—how the translation knowledge we find in bilingual term lists can be used.

Pirkola observed that the distinction between different query terms and different translations of the same query term should be recognized in the structure of a translated query. Specifically, he suggested treating the translations of a query term as if they were synonyms, demonstrating this by using InQuery's synonym operator (#syn) to group alternate translations and InQuery's weight averaging operator (#sum) to combine the weights from each synonym set into document scores [9]. Pirkola's initial experiments were performed using English queries and Finnish documents; similar results are now available for a broad array of language pairs (c.f., [1, 6]). In this paper, we present an alternative to structured queries that achieves a similar effect at indexing time.

The approach to structured query formulation that Pirkola introduced raises two important issues that limit the range of scenarios to which it can be applied:

- The computation required by the InQuery synonym operator is complex, so queries using that operator will be much slower than other approaches to query translation if several alternative translations are known for many of the query terms [5].
- InQuery was designed for commercial applications, so the source code is not available. This limits the ability of researchers to explore variants of the synonym operator that might be better tuned to CLIR applications.

We have addressed these limitations by implementing a computation that closely approximates to that performed by InQuery's synonym operator at indexing time using the freely available MG information retrieval system [11]. In the remainder of this paper we describe the computation performed by the synonym operator in Pirkola's

structured query technique, describe our indexing-time implementation, present the results of an experiment to assess the effectiveness and efficiency of our implementation, and identify opportunities for extending this line of research in the future.

2 Structured Queries

The key idea in so-called "bag-of-terms" information retrieval systems such as InQuery is to compute a weight for each term in every document, and then combine the weights for each query term on a document-by-document basis in order to compute a score for each document. These scores can then be used to rank the available documents in order of decreasing likelihood that they satisfy the information need expressed by the query. The computation of term weights can be based on three principal sources of evidence:

Within-document term frequency Term frequency is the count of the number of occurrences of a given term in a given document, or some monotone function of that count. TF provides a measure of the relative importance of the given term with respect to other terms in the same document. The location of a term within a document can be used to bias the weight given to that term. For example, words appearing in the headline of a news story might receive greater weight, while words that appear in the undisplayed author-assigned metadata fields of a Web page might receive less weight (since they often contain "spam" terms).

Across-document collection frequency Collection frequency is the count of the number of documents in which a term appears. It is a measure of the degree of specificity of the term with respect to the collection. The most common form of collection frequency measure is Inverse Document Frequency (IDF), which is an information content measure that reflects the degree of surprise associated with finding that a document contains the term.

Length The length of a document representation is used to normalize the contribution of the first two sources of evidence in a way that facilitates cross-document comparisons. In its simplest form, length might be measured as the number of terms in the document (the sum of the term frequencies), but more complex measures that also account for collection frequency are also commonly used (in so-called "vector-space" systems).

These sources of evidence are typically used to compute the weight of each term in each document in a way that rewards high term frequencies, low collection frequencies, and short lengths.

In CLIR applications, the query and the document use terms from different languages, so some form of translation is needed. The effect of the InQuery synonym operator in Pirkola's structured query method is to compute query-language term weights based on document-language evidence as follows:

$$TF_j(Q_i) = \sum_{\{k|\, Q_i \in T(D_k)\}} TF_j(D_k) \tag{1}$$

$$CF(Q_i) = |\bigcup_{\{k|\, Q_i \in T(D_k)\}} \{d|\, D_k \in d\}| \tag{2}$$

$$L_k = L_k \tag{3}$$

where Q_i is a query-language term, D_k is a document-language term, $TF_j(X)$ represents the number of occurrences of term X in document j, $CF(X)$ represents the number of documents that contain term X, $T(D_k)$ is the set of query-language translations for document-language term D_k, d is a document, and L_k is the length measure of document k, computed in the same way it would have been if document-language terms were being indexed [3].[1] The effect of these equations is to treat every translation as equally likely and separately estimate the term frequency, document frequency, and length in a manner similar to the way those parameters are computed when stemming is used in a monolingual context. Specifically, the term frequency is the sum of the term frequencies of any possible translation, the collection frequency is the number of documents that contain any translation, and the computation of the length is unchanged. The comparison with stemming is easily seen if "token" is substituted for "translation" in the prior sentence. It is this analogy which motivates our design of an indexing-time analogue for structured queries in the next section.

This way of using document-language evidence has the net effect of suppressing the weight of query-language terms that are associated through translation with *any* common document-language term (i.e., one that appears in many documents). A brief examination of each formula will help to explain why this occurs. The dominant effect results from the CF formula, which can produce a result no smaller than the CF of the most common contributing document-language term. For example, the Spanish term "conducir" is related through translation to the English terms "fly," "go," "pilot," and "drive." Since "go" appears in a great many English documents, "conducir" would receive a high CF and thus a lower term weight if used in a Spanish query.

By contrast, because TF is a within-document measure, the effect of the summation on TF is more often helpful than harmful. Consider the case of the English term "fly," with is related through translation to the Spanish terms "mosca" (a type of insect), "volar" (to travel by airplane), and "conducir" (to pilot an aircraft). Spanish documents that contain "conducir" might also contain "volar." In such cases, summing the term frequencies could produce a beneficial effect by combining the contributions of topically related terms. By contrast, since documents about airplanes rarely mention insects, the set of Spanish documents that contain "mosca" is unlikely to contain either of the other two terms. The few cases in which unrelated translations do occur in the same document will indeed have the effect of giving a query-language term more weight than it deserves, but such cases are likely to be sufficiently rare to have little net effect on retrieval results.

It is computationally expensive to compute term weights in this way at query time because the postings file must be traversed to compute the union in equation 2. The time required to perform this computation increases with both the number of translations for each term, and with the number of documents in which each translation is found. In an earlier study we found that structured queries required about 8 times longer than a corresponding monolingual query [7], although that factor undoubtedly varies with the number of translations that are known for each query term. It is equation 2 that is

[1] InQuery actually computes the sum and the union over the document-language translations of the query terms, but because bilingual term lists can be thought of as a set of reversible translation pairs, our formula is equivalent.

responsible for this delay, since computing the union requires that access to the postings file. Since the postings file is typically so large that it must be stored on disk, the number of disk accesses that are required to process each query is increased.

A more efficient variant of structured queries has been implemented in the Queens College PIRCS system [4]. In that implementation, the union in equation 2 is replaced by:[2]

$$CF(Q_i) = \sum_{\{k | Q_i \in T(D_k)\}} CF(D_k) \tag{4}$$

where Q_i is a query-language term, D_k is a document-language term, $CF(X)$ represents the number of documents that contain term X, $T(D_k)$ is the set of query-language translations for document-language term D_k, d is a document. This formula computes the sum of document frequencies of each term in the query. The document frequency of each term is the number of documents containing a term whose one of the translations is the query term.

If near-synonyms rarely occur among the translations used in the dictionary, this equations 2 and 4 will compute similar values. With a richer dictionary that contains more near-synonyms, equation 4 would tend to overestimate the collection frequency if the near-synonyms often occur within the same document. We are not aware of any experiments in which this approach has been compared with the computation that is implemented by InQuery's synonym operator.

3 Translation-Based Indexing

The goal of the indexing stage in an information retrieval system is to preprocess the document collection to create an index structure that can be efficiently searched to obtain a value (known as a "term weight") for each document that contains a query term. If stemming will be used at query time in a monolingual system, then it is the term weights associated with stems (rather than surface forms) that would normally be indexed. In a CLIR system, the natural extension of this idea is to index the term weights associated with translations (or, if queries will be stemmed, the weights associated with stems of translations). The key question is therefore how such term weights should be computed. This is the focus of translation-based indexing.

Translation-based indexing requires access to a machine readable bilingual dictionary (or some other form or translation lexicon) in which the source language is the language in which the documents are written and the target language is the language in which the queries will be posed. The key idea is simply to index every possible translation of each document-language term.

We modified the August, 1999 release of the Managing Gigabytes (MG) system (mg-1.2.1) to incorporate translation-based indexing.[3] The changes were localized to the inversion steps in the first and second pass of the indexing process. In each case,

[2] Personal communication with K.L. Kwok.

[3] Source code for MG is available under the GNU public license at http://www.cs.mu.oz.au/mg/ and our modifications are available at http://tides.umiacs.umd.edu.

we replaced each document-language word with all of its target (query) language translations. Since there are often several translations of a single term, the second pass (in which the postings file is built) results in more disk accesses, and hence slower indexing, when translation-based indexing is used. The expected slowdown is:

$$t_c/t_m > c * f \tag{5}$$

where t_c is the time required for translation-based indexing, t_m is the time required for document-language indexing, c is the fraction of the terms for which a translation is known (the "by-token" coverage of the dictionary with respect to the collection being indexed), and f is the "fanout" of the dictionary, the average number of translations that are known for each term. The expected indexing time is somewhat greater than the right hand side of the formula would indicate because fanout is normally computed on a by-type basis, giving common terms (which typically have more translations) the same weight as rare terms. We also made some minor modifications to MG to accommodate languages other than English.

MG's implementation of vector space retrieval systems perform length normalization in a manner different from InQuery's inference network model. In InQuery, document length is incorporated in weight computations by computing a ratio between the term frequency and the document length. Equation 3 therefore results in an appropriate computation with document-language terms. MG, by contrast, normalizes for document length in a way that further increases the relative weight of terms with low collection frequencies. This is accomplished using cosine normalization as follows:

$$w'_{j,k} = \frac{w_{j,k}}{\sqrt{\sum_k w_{j,k}^2}} \tag{6}$$

where $w'_{j,k}$ is the normalized weight for term j in document k and $w_{j,k}$ is the corresponding weight before length normalization. This difference in length normalization strategies precludes a straightforward analytical comparison between structured queries and translation-based indexing, so we have conducted some experiments to characterize the effect.

4 Evaluation

We performed a preliminary evaluation to characterize the efficiency and effectiveness of our implementation. We used the 161 MB collection of 44,013 1994 French Le Monde news articles from the Cross-Language Evaluation Forum (CLEF-2000) collection. For each of the 40 topics, we formed three queries: short (all words from the title field), medium (all words from the title and description fields), and long (all words from the title, description and narrative fields).

Figure 1 shows the effect of adding translation-based indexing on the time required to index a collection in MG on a 750 MHz Sun Blade workstation with 1 GB of physical memory. We used a French-English dictionary (referred to as "Dict1" in Table 1) that has an average of 2.1 English translations per French term by type (25,037 unique French terms, 52,475 English translations) and 85% by-token coverage of the French

document collection (14.2 million / 16.6 million tokens). The observed effect on indexing time is consistently just over a factor of two, which matches well with our expectations ($t_c/t_m > 0.85 * 2.1$).

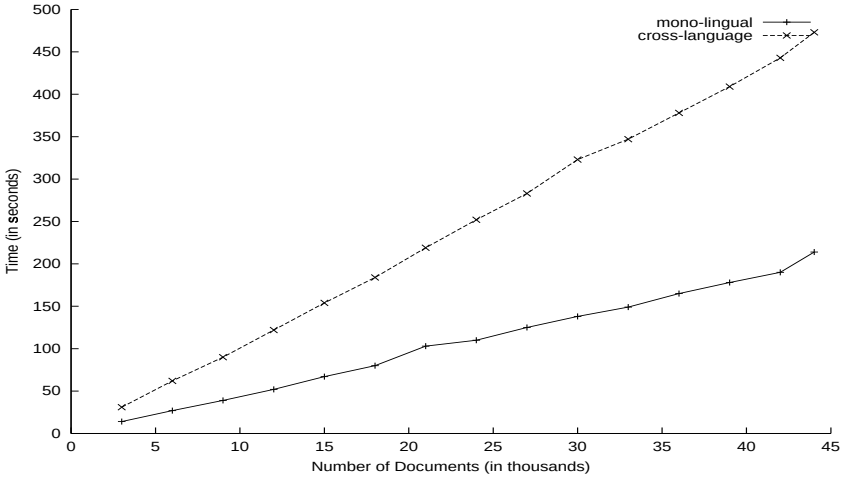

Fig. 1. Indexing time (in seconds) with and without translation-based indexing.

The effectiveness results in Table 1 also behave about as we would expect, with mean uninterpolated average precision increasing with query length from 0.10 (for title queries) to 0.13 (for title/description/narrative queries). These results are about one-third of the mean average precision obtained by MG in a monolingual condition (with French queries; 0.26 and 0.31, respectively), and are consistent with results obtained when using structured queries with InQuery. The relatively poor cross-language performance in this case reflects some deficiencies in our initial implementation (e.g., we did not remove accents when retaining untranslated terms, and we tried no morphological variants if the surface form of a French term was not found in the dictionary) and the relatively poor coverage of the dictionary that we chose. To partially characterize the effect of dictionary coverage, we reran both systems with a second French-English dictionary and achieved a somewhat better results from translation-based indexing (0.14 and 0.15, respectively). From this we conclude that our present implementation of translation-based indexing achieves results that are similar to those achieved by a comparable implementation of structured queries.

5 Future Work

Clearly, the next thing that we need to do is compare the relative performance of translation-based indexing under a broader range of conditions (bilingual dictionaries and test collections), and with a broader set of contrastive conditions (e.g., balanced

		InQuery Dict1	MG Dict1	InQuery Dict2	MG Dict2
Monolingual	T	0.29	0.26	0.29	0.26
	TD	0.33	0.28	0.33	0.28
	TDN	0.36	0.31	0.36	0.31
Structured Queries	T	0.10		0.10	
	TD	0.12		0.12	
	TDN	0.13		0.13	
Translation-Based Indexing	T		0.10		0.14
	TD		0.11		0.14
	TDN		0.13		0.15

Table 1. Uninterpolated mean average precision for different query lengths (T=title queries, TD=title/description queries, TDN=title/description/narrative queries).

translation [6] and the PIRCS variant of structured queries). As part of this effort, we intend to integrate additional features such as orthographic normalization for untranslated terms, phrase translation, and backoff translation [10], all of which are known to improve retrieval effectiveness.

The existence of a freely available system for translation-based indexing will also make it possible to explore several other potentially promising lines of inquiry:

- Post-translation resegmentation. Term translation sometimes yields multiword expressions, but it is well known from monolingual retrieval experiments that indexing the constituent words of a multiword expression can be beneficial. In the context of translation-based indexing, this creates a credit assignment problem in which the weight computed for a multiword expression provides a basis for computing the weights of the constituent words.
- Context-sensitive translation. If sharp syntactic and semantic constraints are available, the set of possible translations for the same document-language term could be varied based on this evidence. Queries, which are often short, typically offer less scope for the this sort of analysis, so context-sensitive approaches would naturally favor a document translation architecture.
- Weighted summation for term frequency. If the relative likelihood of alternative translations is known, the contribution of each translation to the sum operator could be weighted appropriately.
- Proportional representation for collection frequency. If the relative likelihood of alternative translations is known, the contribution of each translation to the union operator could be apportioned appropriately.
- Multilingual indexing. Translation-based indexing could conceivably be extended to support multiple query languages by using a merged bilingual term list to identify the translations in each language that should be indexed. Open questions regarding this approach include whether the occurrence of the same string with different meanings in two query languages would adversely affect retrieval effectiveness, and whether present approaches to document length normalization would need to be adapted to accommodate the richer document representations.

6 Conclusion

Translation-based indexing offers a new capability, but like many new ideas it augments, rather than replaces, what came before. When only a single query language must be supported, translation-based indexing offers a way of achieving a substantial reduction in query execution time without adversely affecting retrieval effectiveness. Structured queries, by contrast, offer greater flexibility at query time, both because users can potentially help in the translation process (c.f. [8]), and because a broad range of query languages can easily be supported. Perhaps the most lasting contribution of this work, however, will be the availability of a freely available implementation of translation-based indexing in a state-of-the-art retrieval retrieval system—something that we hope will inspire further work along the lines outlined above.

Acknowledgments

The authors are grateful to Gina Levow and Clara Cabezas for assistance with the implementation of balanced translation, Dina Demner-Fushman for providing dictionary coverage statistics, and Jianqiang Wang for his help with InQuery. This work has been supported in part by DARPA cooperative agreement N660010028910.

References

1. Lisa Ballesteros and W. Bruce Croft. Resolving ambiguity for cross-language retrieval. In C.J. Van Rijsbergen W. Bruce Croft, Alistair Moffat, editor, *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 64–71. ACM Press, August 1998.
2. Gregory Grefenstette and Julien Nioche. Estimation of emglish and non-english language use on the www. In *Content-Based Multimedia Information Access*, April 2000.
3. Jaana Kekäläinen and Kalervo Järvelin. The impact of query structure and query expansion on retrieval performance. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 130–137, August 1998.
4. K. L. Kwok. NTCIR-2 chinese, cross-language retrieval experiments using PIRCS. In Noriko Kando, editor, *Proceedings of the Second NII Test Collection Information Retrieval Workshop*. 2001. http://ir.cs.qc.edu.
5. Douglas W. Oard, Gina-Anne Levow, and Clara Cabezas. CLEF experiments at the University of Maryland: Statistical stemming and backoff translation strategies. In *Working Notes of the First Cross-Language Evaluation Forum (CLEF-1)*, September 2000. http://www.glue.umd.edu/∼oard/research.html.
6. Douglas W. Oard and Jianqiang Wang. NTCIR-2 ECIR experiments at Maryland: Comparing pirkola's structured queries and balanced translation. In Noriko Kando, editor, *Proceedings of the Second NII Test Collection Information Retrieval Workshop*. 2001. http://www.glue.umd.edu/∼oard/research.html.
7. Douglas W. Oard, Jianqiang Wang, Dekang Lin, and Ian Soboroff. Trec-8 experiments at maryland: Clir, qa and routing. In *Eight Text Retrieval Conference*, November 1999.
8. William C. Ogden and Mark W. Davis. Improving cross-language text retrieval with human interactions. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, January 2000. http://crl.nmsu.edu/∼ogden.

9. Ari Pirkola. The effects of query structure and dictionary setups in dictionary-based cross-language information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 55–63, August 1998.
10. Philip Resnik, Douglas Oard, and Gina Levow. Improved cross-language retrieval using backoff translation. In *First International Conference on Human Language Technologies*, 2001. http://www.glue.umd.edu/∼oard/research.html.
11. Ian H. Witten, Alistair Moffett, and Timothy C. Bell. *Managing Gigabytes*. Morgan Kaufmann, San Francisco, second edition, 1999.